

US010229124B2

(12) **United States Patent**
Oliver et al.

(10) **Patent No.:** **US 10,229,124 B2**
(45) **Date of Patent:** **Mar. 12, 2019**

(54) **RE-DIRECTING TENANTS DURING A DATA MOVE**

(58) **Field of Classification Search**
None
See application file for complete search history.

(71) Applicant: **Microsoft Technology Licensing, LLC**,
Redmond, WA (US)

(56) **References Cited**

(72) Inventors: **David Charles Oliver**, Bellevue, WA (US); **Ming-wei Wang**, Bellevue, WA (US); **Gang Zhao**, Redmond, WA (US); **Daniel Keith Winter**, Monroe, WA (US); **Shyam Narayan**, Redmond, WA (US); **Parul Manek**, Redmond, WA (US)

U.S. PATENT DOCUMENTS

6,145,066 A 11/2000 Atkin
6,195,701 B1 2/2001 Kaiserswerth et al.
(Continued)

FOREIGN PATENT DOCUMENTS

(73) Assignee: **Microsoft Technology Licensing, LLC**,
Redmond, WA (US)

EP 1462960 A2 9/2004
EP 1503313 A1 2/2005
WO 2015043679 A1 4/2015

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 831 days.

OTHER PUBLICATIONS

International Search Report and Written Opinion for International Patent Application No. PCT/US2016/029909, dated Jul. 27, 2016, date of filing: Apr. 29, 2016, 11 pages.

(21) Appl. No.: **14/874,181**

(Continued)

(22) Filed: **Oct. 2, 2015**

Primary Examiner — Anhtai V Tran

(65) **Prior Publication Data**
US 2016/0321276 A1 Nov. 3, 2016

(74) *Attorney, Agent, or Firm* — Christopher J. Volkman; Kelly, Holt & Christenson, PLLC

Related U.S. Application Data

(57) **ABSTRACT**

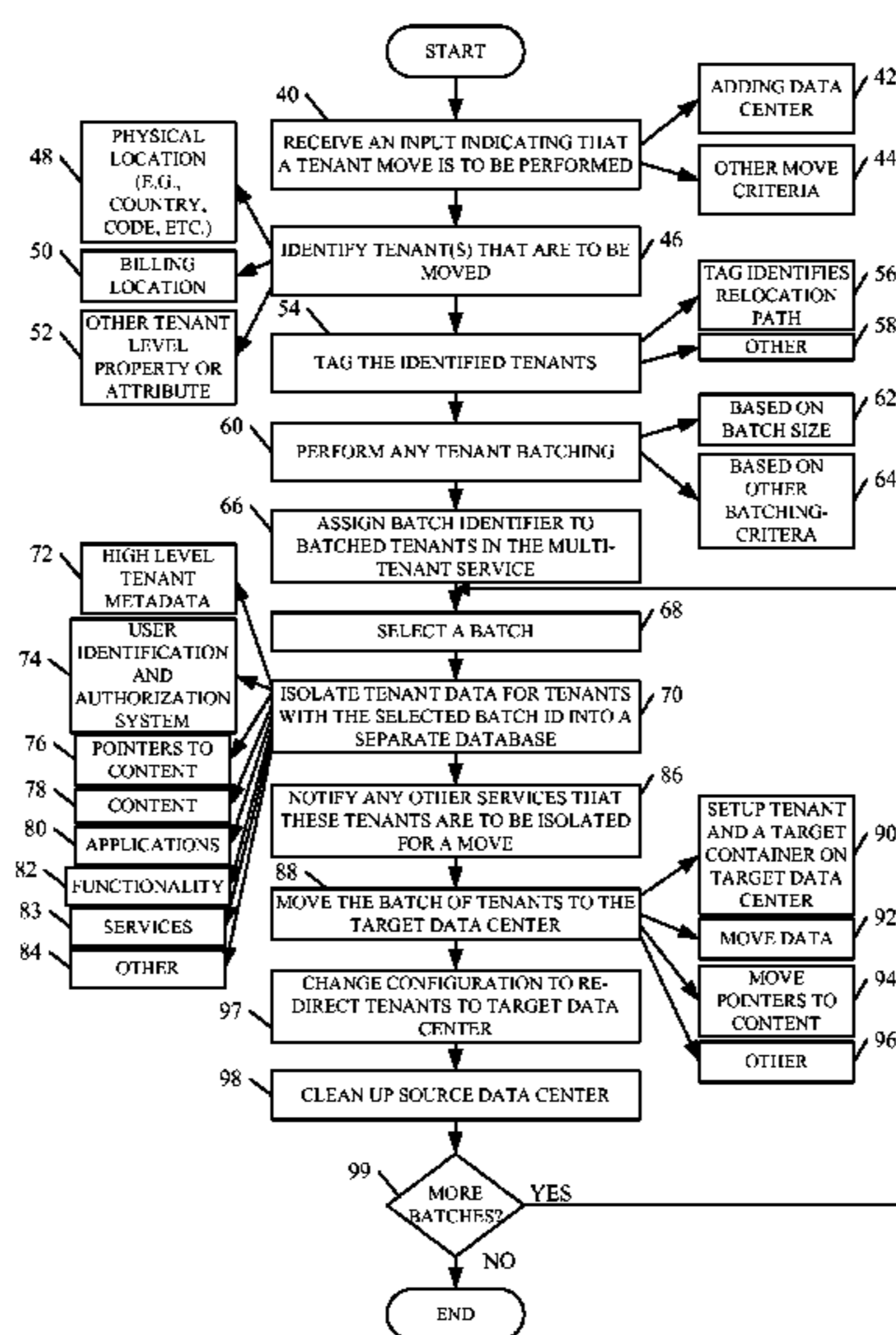
(60) Provisional application No. 62/156,096, filed on May 1, 2015, provisional application No. 62/156,082, filed (Continued)

Data to be moved from a source system to a target system, for a set of tenants, is first identified. The data is enumerated by a first computing instance in the source system to obtain an enumeration list. Data is copied from the source system to the target system based on the enumeration list by a second computing instance. The data in the source and target systems is then enumerated by a third computing instance to determine whether any data is still to be moved and another enumeration list is generated. The data still to be moved is then moved based on the other enumeration list.

(51) **Int. Cl.**
G06F 17/30 (2006.01)
H04L 29/08 (2006.01)

20 Claims, 10 Drawing Sheets

(52) **U.S. Cl.**
CPC **G06F 17/30079** (2013.01); **G06F 17/303** (2013.01); **G06F 17/30575** (2013.01); **H04L 67/02** (2013.01); **H04L 67/2814** (2013.01)



Related U.S. Application Data

on May 1, 2015, provisional application No. 62/156,089, filed on May 1, 2015.

(56)

References Cited

U.S. PATENT DOCUMENTS

6,393,434	B1	5/2002	Huang et al.	
6,405,294	B1	6/2002	Hayter	
7,281,014	B2	10/2007	Toner	
7,523,141	B2	4/2009	Makismenka et al.	
7,669,208	B2	2/2010	Wu et al.	
7,792,794	B2	9/2010	Mills et al.	
7,890,454	B2	2/2011	Hamilto et al.	
8,046,552	B2	10/2011	Larson	
8,156,074	B1	4/2012	Multer et al.	
8,166,101	B2	4/2012	Shah	
8,261,033	B1	9/2012	Slik et al.	
8,352,425	B2	1/2013	Bourbonnais et al.	
8,386,501	B2	2/2013	Cahill et al.	
8,805,783	B2	8/2014	Muhunthan et al.	
8,924,352	B1	12/2014	Andruss et al.	
8,996,675	B2	3/2015	Kalyanaraman et al.	
2004/0039962	A1	2/2004	Ganesh et al.	
2005/0027747	A1	2/2005	Wu et al.	
2005/0027817	A1	2/2005	Novik et al.	
2007/0094308	A1	4/2007	Mitchell et al.	
2010/0076933	A1	3/2010	Hamilton et al.	
2010/0082920	A1	4/2010	Larson	
2010/0268788	A1	10/2010	Arimilli et al.	
2011/0321041	A1	12/2011	Bhat et al.	
2012/0042033	A1	2/2012	Ayala et al.	
2012/0102067	A1	4/2012	Cahill et al.	
2012/0173483	A1*	7/2012	Hartig	G06F 9/4856 707/610
2012/0174085	A1*	7/2012	Driesen	G06F 8/65 717/168
2012/0233329	A1	9/2012	Dickinson et al.	
2013/0085998	A1	4/2013	Barker et al.	
2013/0179883	A1	7/2013	Hayward et al.	
2013/0297800	A1	11/2013	Considine et al.	
2013/0312117	A1	11/2013	Sapp et al.	
2013/0339420	A1	12/2013	Srinivasan et al.	
2013/0346619	A1	12/2013	Panuganty et al.	
2014/0059232	A1*	2/2014	Plattner	H04L 67/1029 709/226
2014/0067758	A1	3/2014	Boldyrev et al.	
2014/0067872	A1*	3/2014	Kruglick	G06F 17/30327 707/797
2014/0114913	A1*	4/2014	Engelko	G06F 17/30079 707/609
2014/0325640	A1	10/2014	Aggarwal et al.	
2015/0020044	A1	1/2015	Venkatesan et al.	
2015/0096011	A1	4/2015	Watt	
2015/0199243	A1	7/2015	Wu et al.	
2016/0321248	A1	11/2016	Narayan et al.	
2016/0321274	A1	11/2016	Narayan et al.	

OTHER PUBLICATIONS

“4—Partitioning Multi-Tenant Applications”, Retrieved From <https://web.archive.org/web/20121216001038/https://msdn.microsoft.com/en-us/library/hh534477.aspx>, Dec. 16, 2012, 36 Pages.

“AppZero’s Cloud Migration Unlocks Production Apps”, Retrieved from <https://web.archive.org/web/20150214081345/https://www.appzero.com/cloud>, Aug. 14, 2014, 3 Pages.

“Non-Final Office Action Issued in U.S. Appl. No. 14/854,761”, dated Apr. 25, 2018, 17 Pages.

Barker, et al., “Cut Me Some Slack: Latency-Aware Live Migration for Databases”, In Proceedings of the 15th International Conference on Extending Database Technology, Mar. 27, 2012, 12 Pages.

Barker, et al., “Shuttle DB: Database-Aware Elasticity in the Cloud”, In Proceedings of the 11th International Conference on Autonomic Computing, Jun. 18, 2014, 11 Pages.

Bowerman, Guy, “Azure Data Center Migration Just Got Easier”, Retrieved From <https://azure.microsoft.com/en-in/blog/azure-data-center-migration-just-got-easier/>, Jan. 12, 2015, 4 Pages.

Clark, et al., “Live Migration of Virtual Machines”, In Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation, vol. 2, May 2, 2005, 14 Pages.

Das, et al., “Live Database Migration for Elasticity in a Multitenant Database for Cloud Platforms”, In Publication of UCSB Computer Science Technical Report, Sep. 2010, 14 Pages.

Elmore, et al., “Zephyr Live Migration in Shared Nothing Databases for Elastic Cloud Platforms”, In Proceedings of the ACM SIGMOD International Conference on Management of Data, Jun. 12, 2011, 12 Pages.

Ghorbani, et al., “Transparent, Live Migration of a Software-Defined Network”, In Proceedings of the ACM Symposium on Cloud Computing, Nov. 3, 2014, 13 Pages.

Igbe, Damian, “How to Migrate an Instance with Zero Downtime: Openstack Live Migration with KVM Hypervisor and NFS Shared Storage”, Retrieved from <https://www.mirantis.com/blog/tutorial-openstack-live-migration-with-kvm-hypervisor-and-nfs-shared-storage/>, Oct. 25, 2013, 41 Pages.

Keller, et al., “Live Migration of an Entire Network (and its Hosts)”, Retrieved From <http://ftp.cs.princeton.edu/reports/2012/926.pdf>, May 2012, 14 Pages.

Kolekar, et al., “Defining a Routing Architecture for Multi-Tenant Virtual Data Centers to Support Host Migration”, In Final Paper of TLEN 5710-Capstone, Apr. 25, 2014, 26 Pages.

Michael, et al., “Database Live Migration with Oracle Multitenant and the Oracle Universal Connection Pool on Oracle Real Application Cluster (RAC)”, In Oracle White Paper, Oct. 2014, 12 Pages.

Michael, “Downtime-Free Live Migration in a Multitenant Database”, In Proceedings of 6th TPC Technology Conference on Performance Evaluation & Benchmarking, Sep. 1, 2014, 25 Pages. “International Search Report and Written Opinion Issued in PCT Application No. PCT/US2016/029911”, dated Aug. 2, 2016, 11 Pages.

“Second Written Opinion Issued in PCT Application No. PCT/US2016/029911”, dated Nov. 17, 2016, 6 Pages.

“International Preliminary Report on Patentability Issued in PCT Application No. PCT/US2016/029917”, dated Feb. 14, 2017, 7 Pages.

“International Search Report and Written opinion issued in PCT Application No. PCT/US2016/029917”, dated Jul. 27, 2016, 14 Pages.

“Second Written Opinion Issued in PCT Application No. PCT/US2016/029917”, dated Nov. 9, 2016, 6 Pages.

Reed, D. P., “Naming and Synchronization in a Decentralized Computer System”, In Technical Report, Sep. 1978, 2 pages.

Second Written Opinion for International Patent Application No. PCT/US2018/029909, dated Nov. 17, 2016, date of filing: Apr. 29, 2016, 7 pages.

* cited by examiner

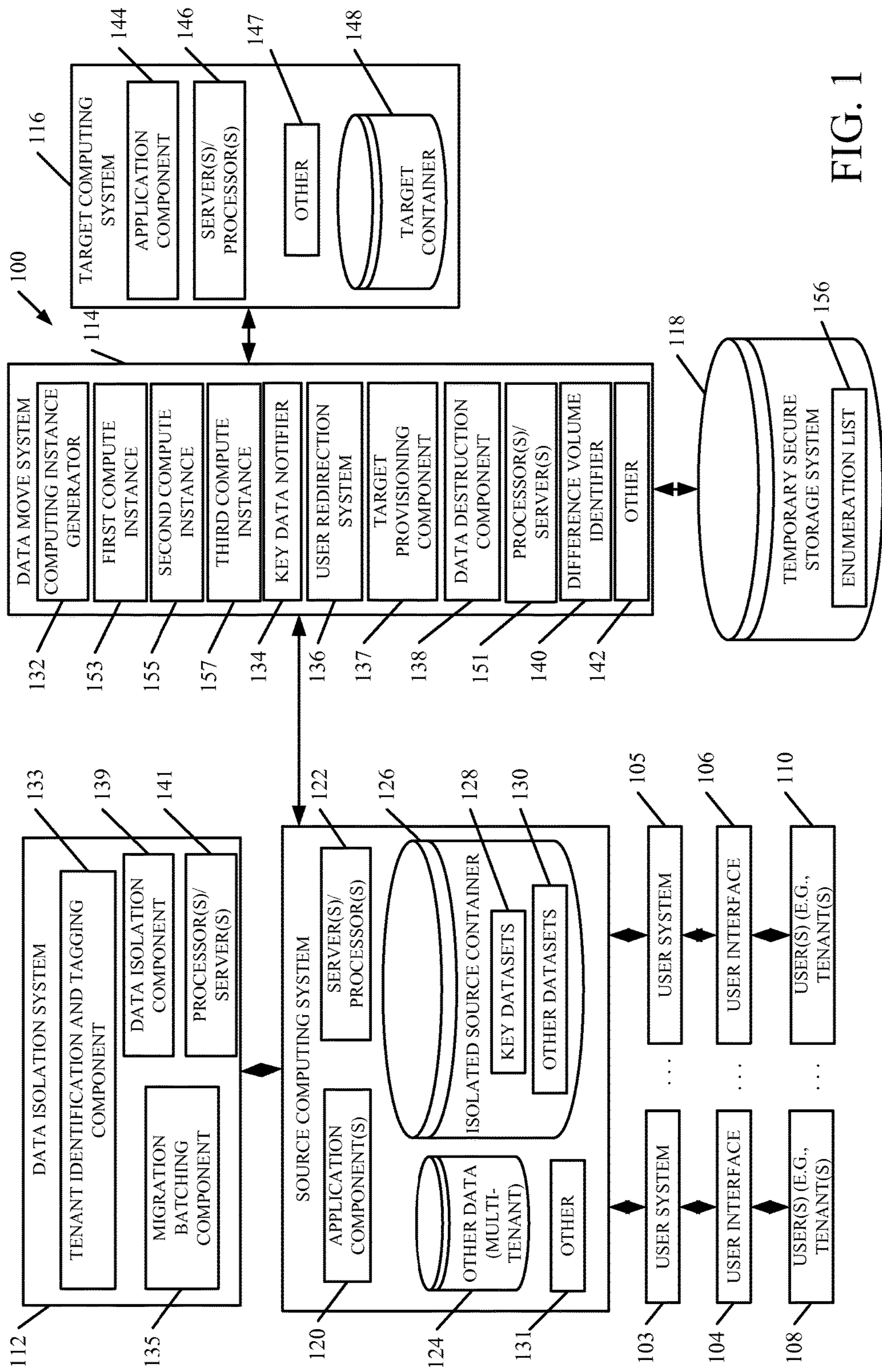


FIG. 1

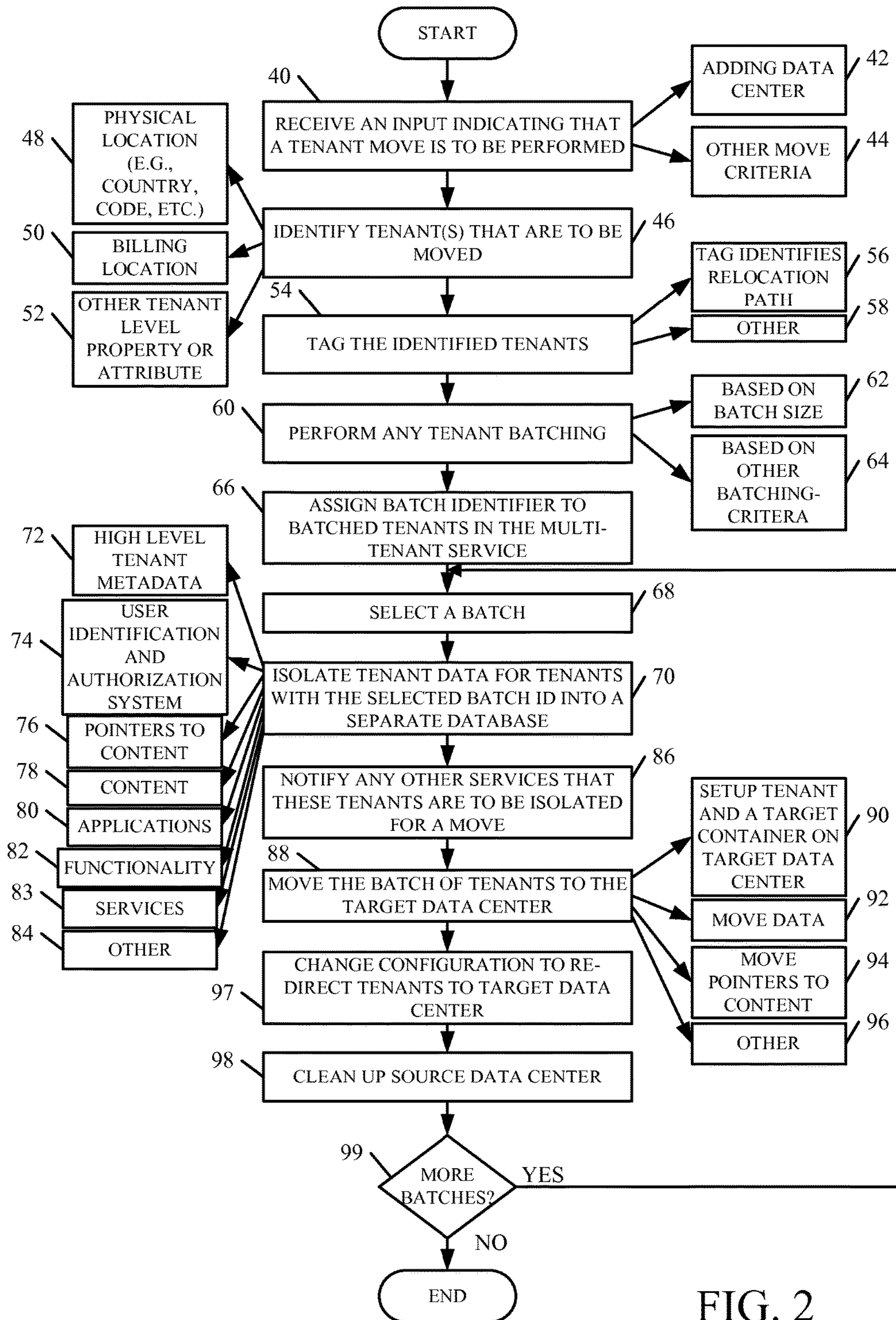


FIG. 2

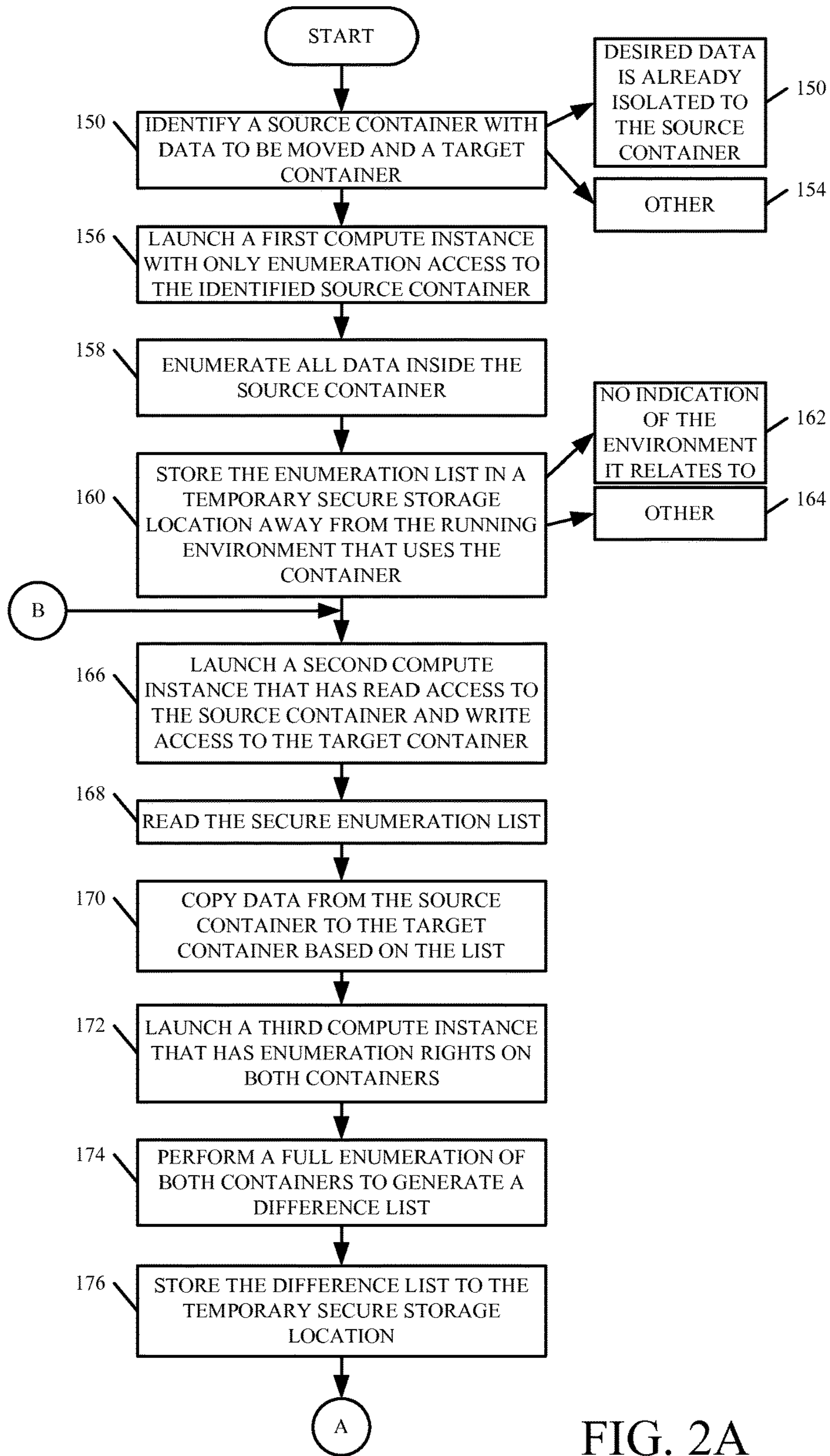


FIG. 2A

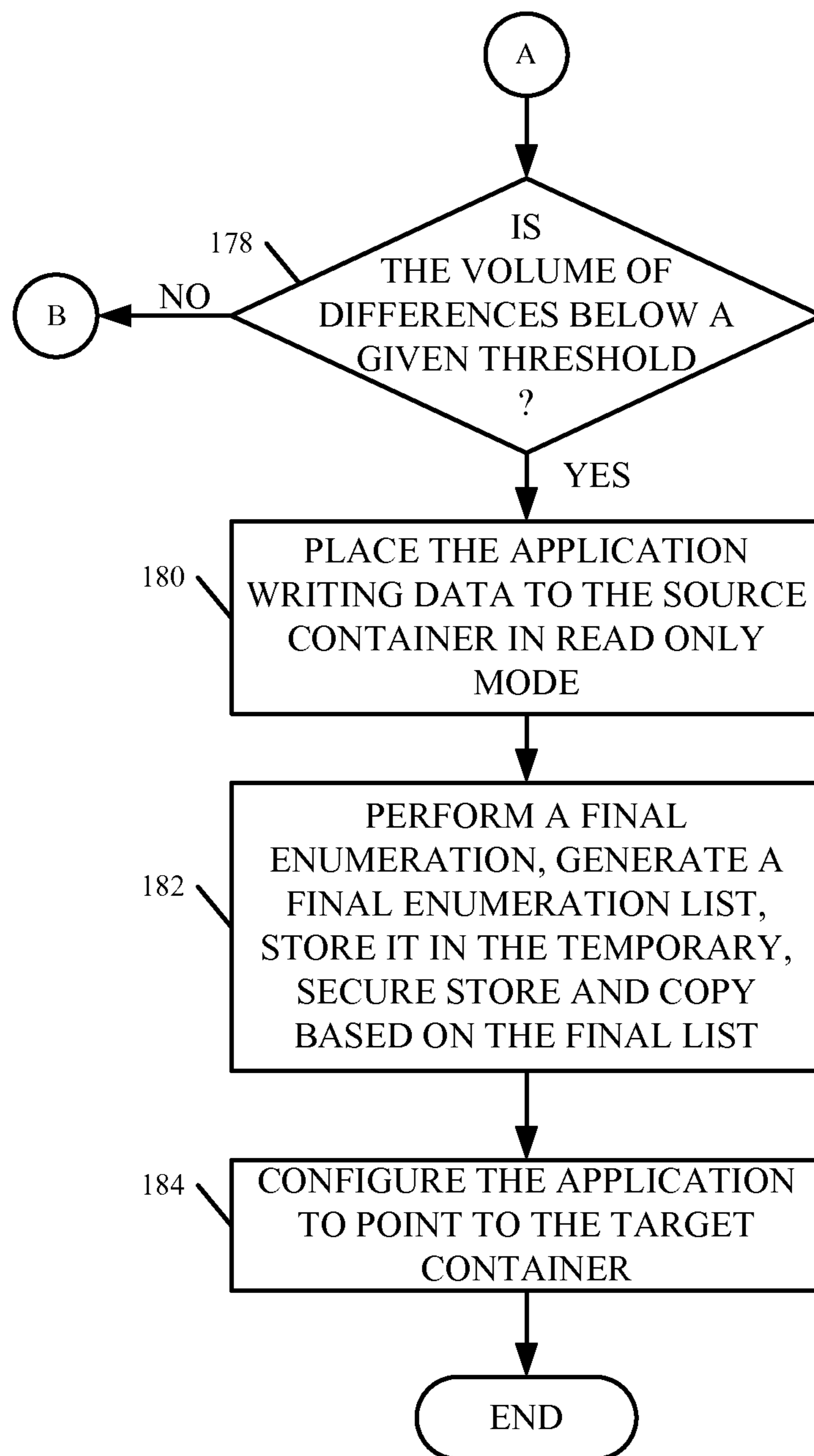


FIG. 2B

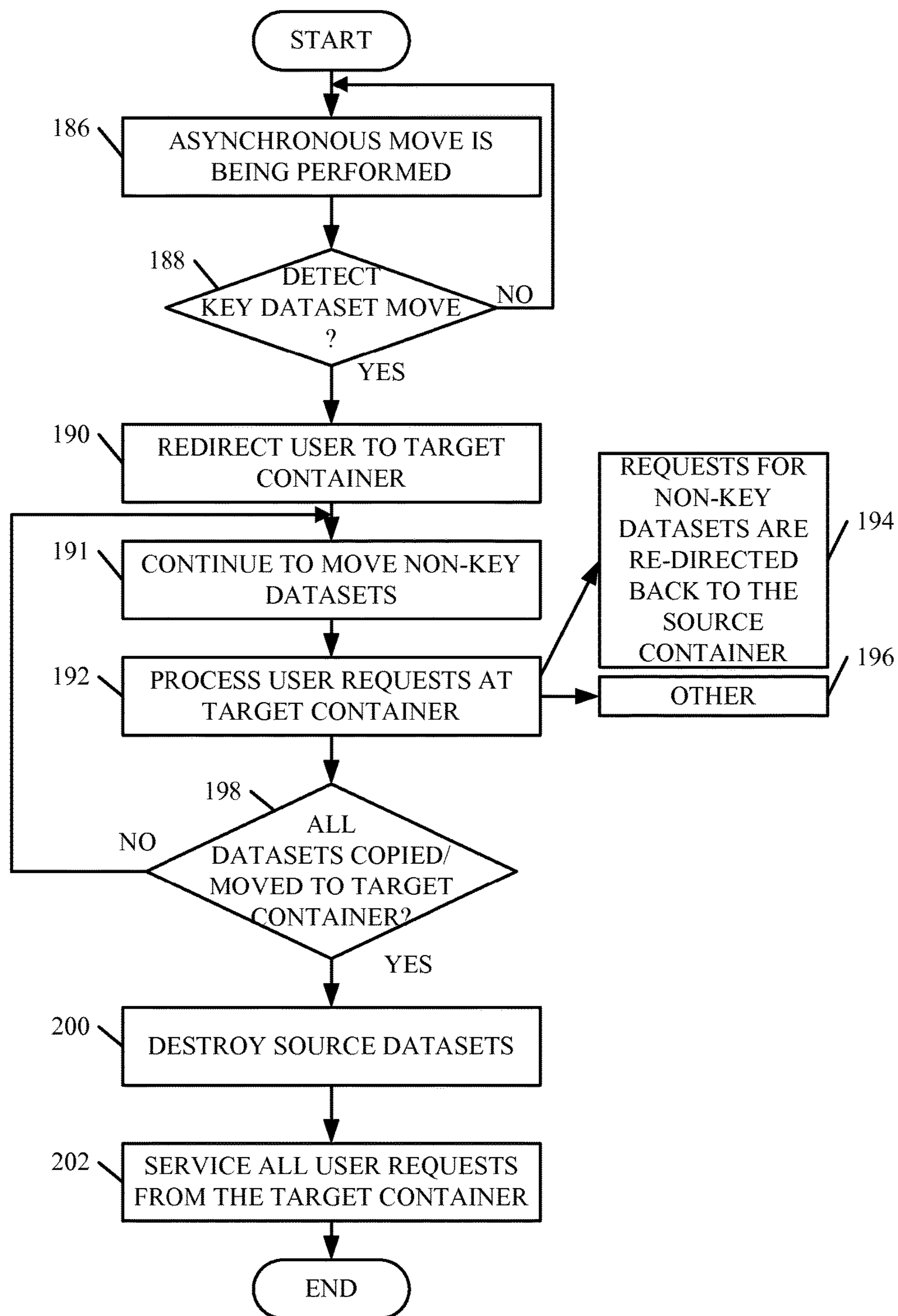


FIG. 3

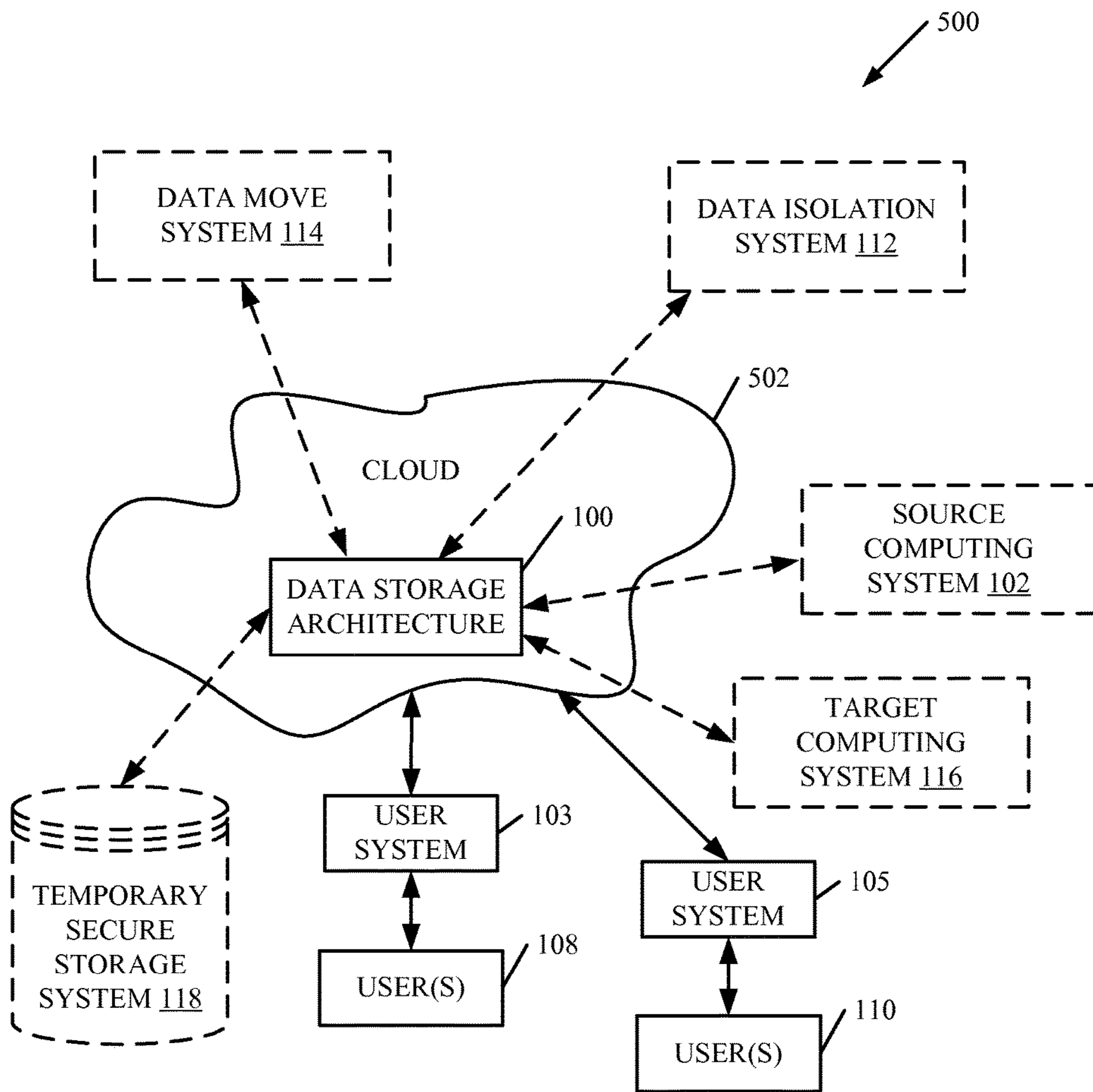


FIG. 4

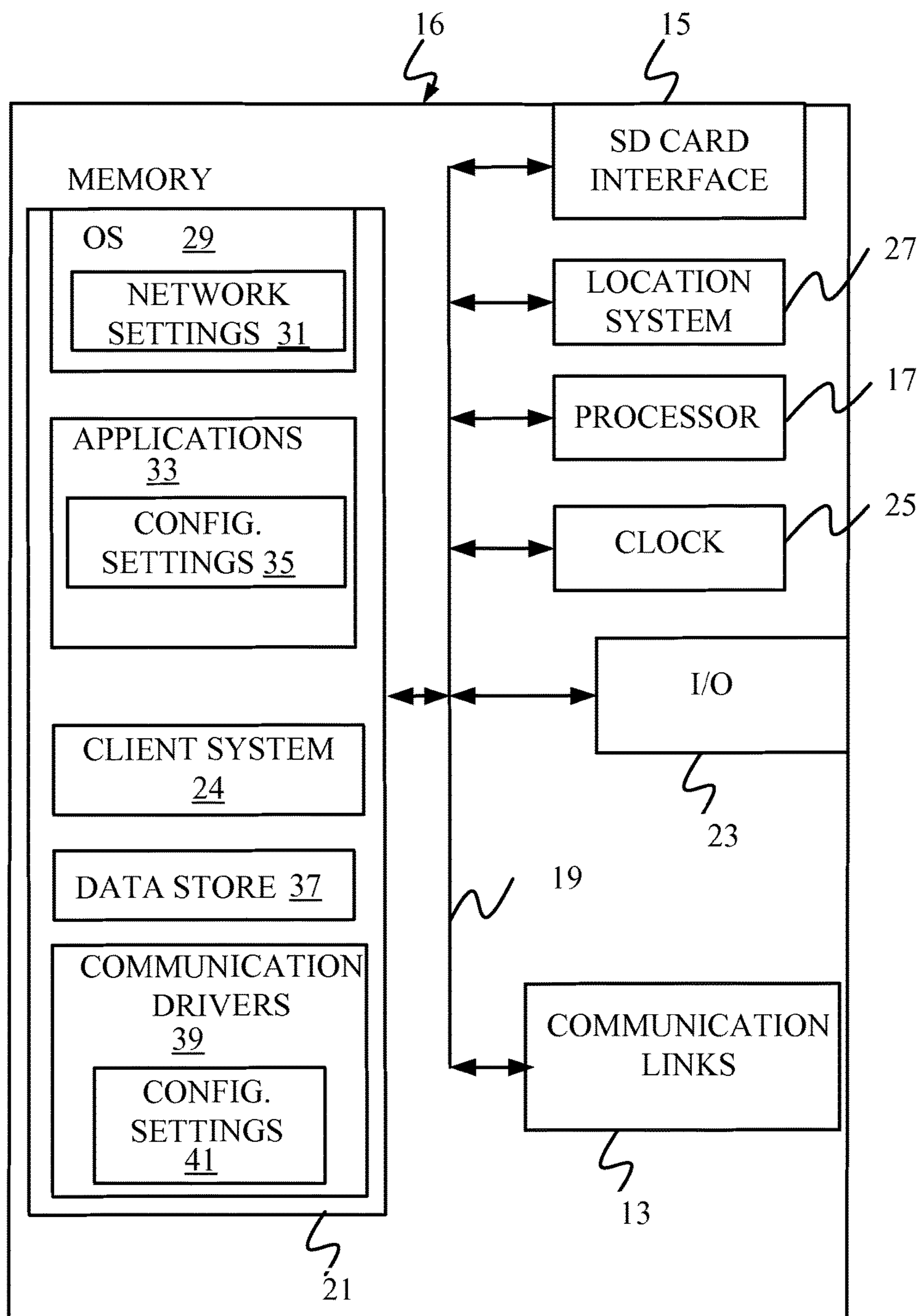


FIG. 5

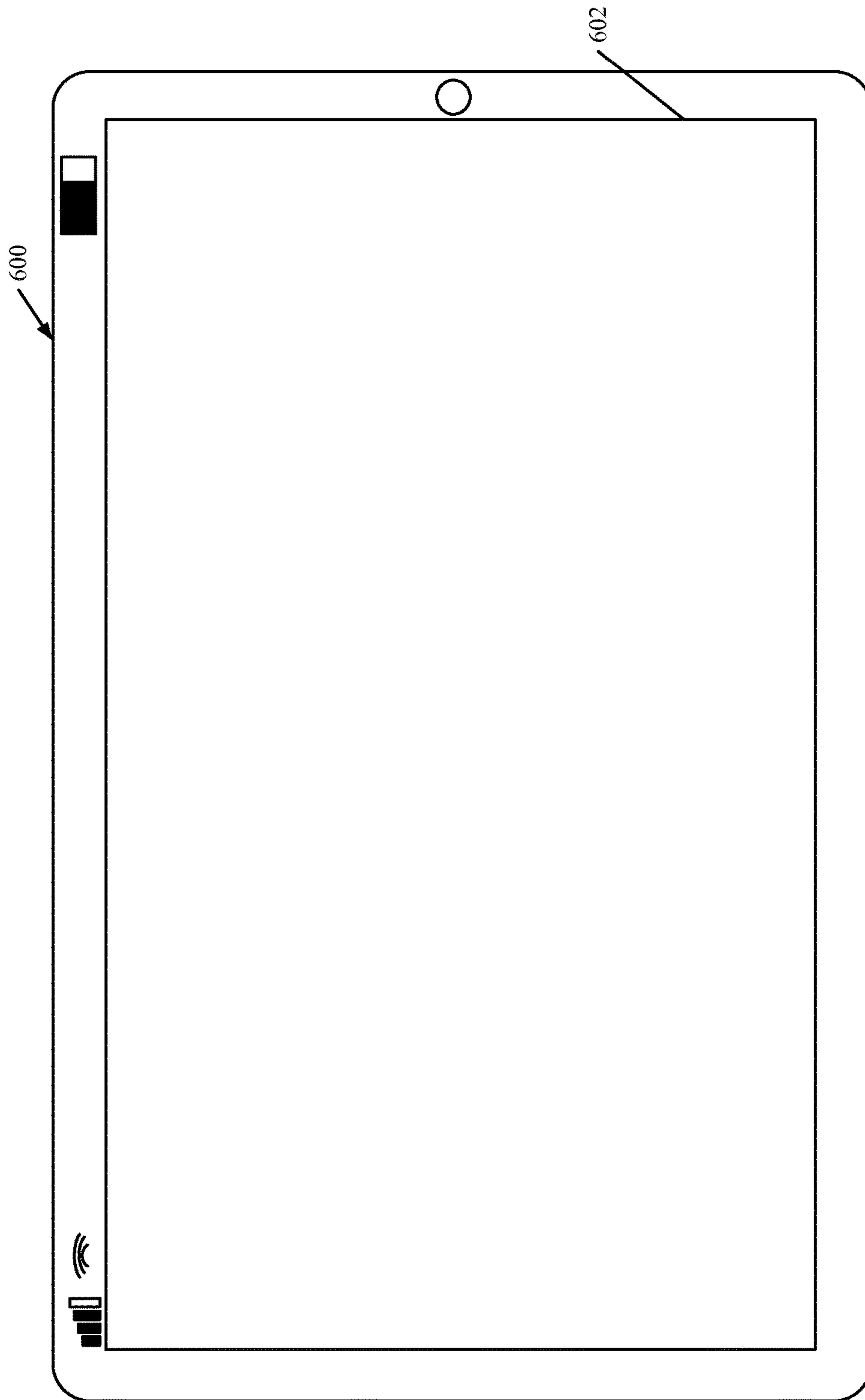


FIG. 6

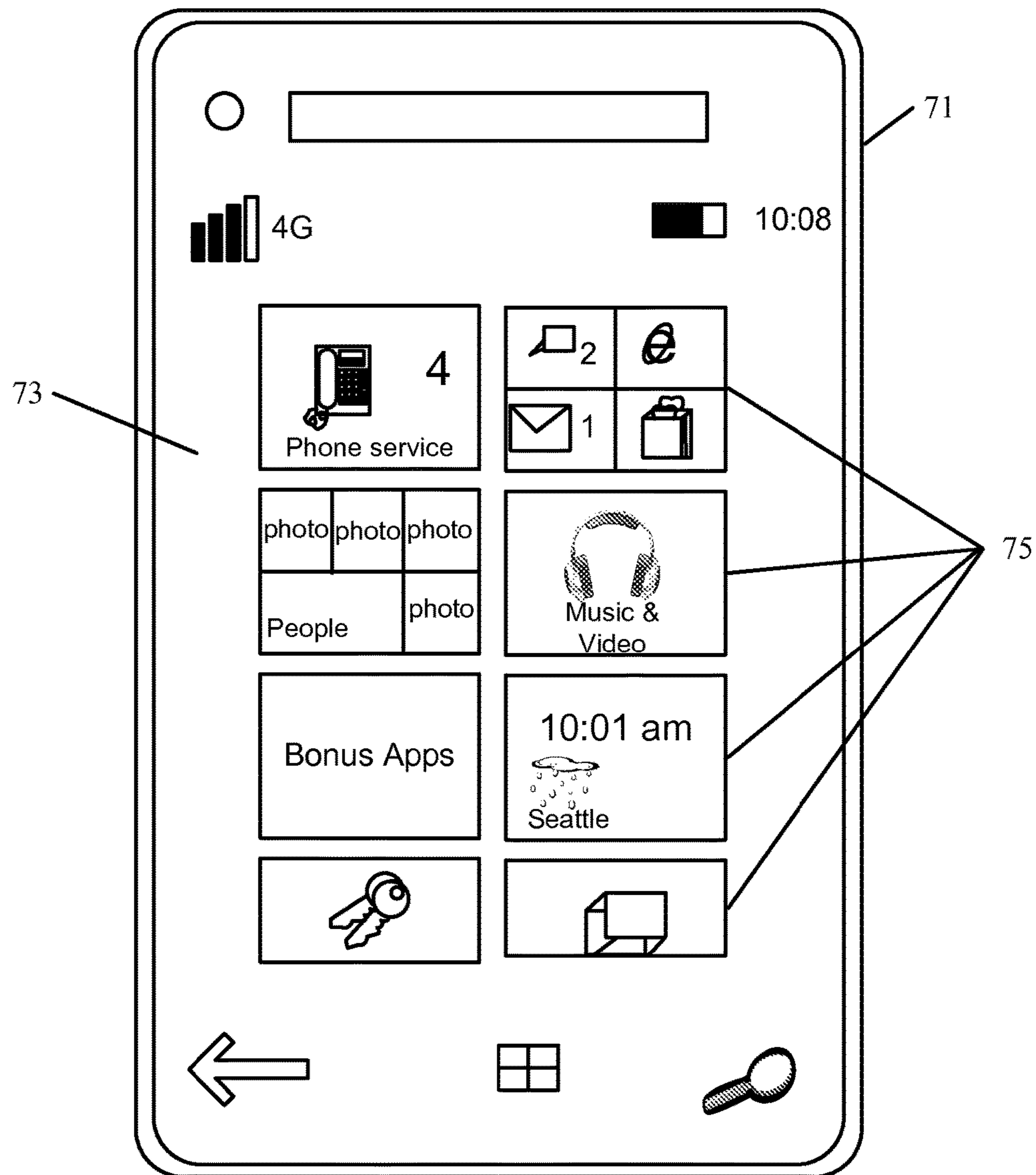


FIG. 7

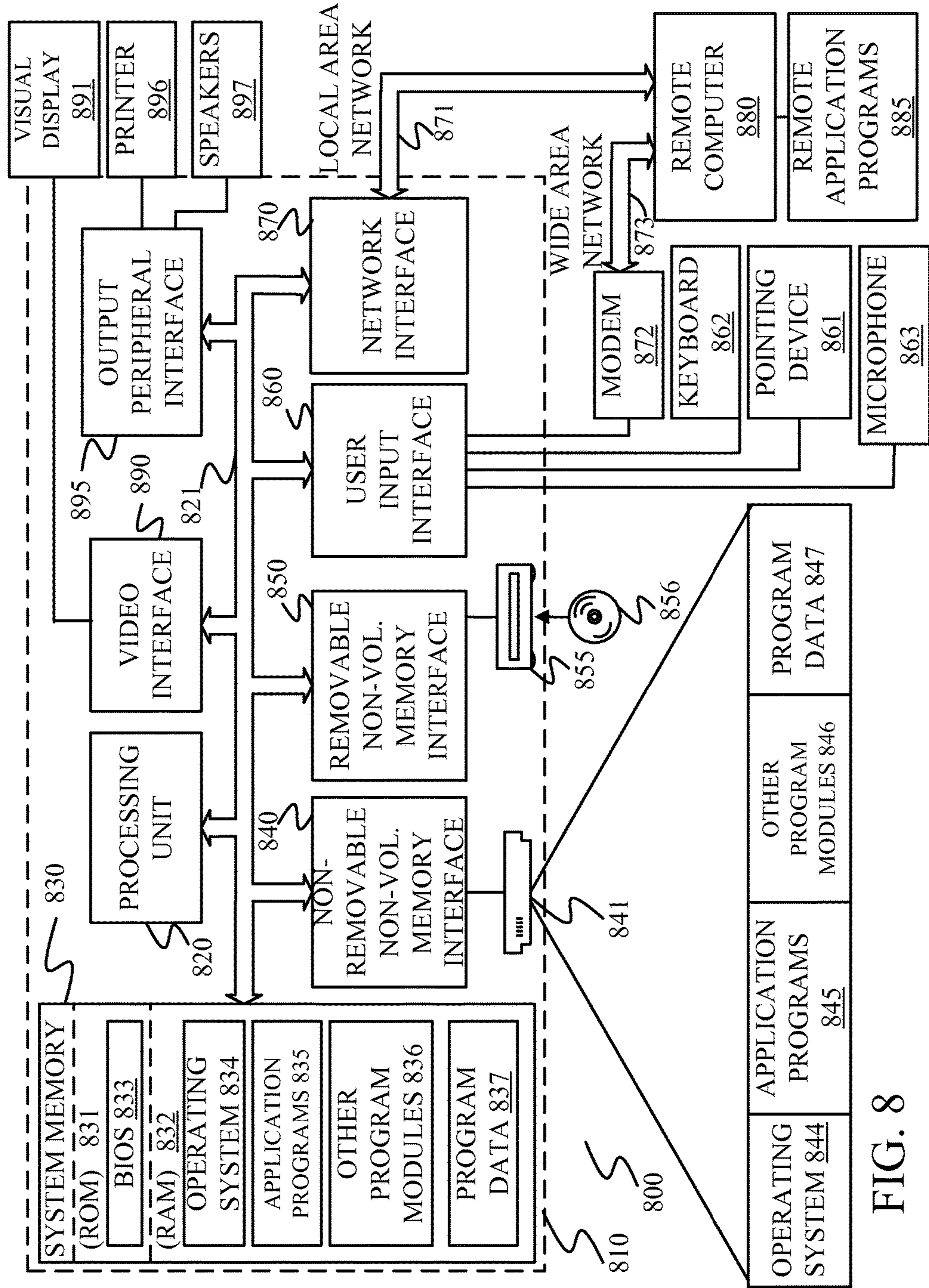


FIG. 8

1**RE-DIRECTING TENANTS DURING A DATA MOVE****CROSS-REFERENCE TO RELATED APPLICATION**

The present application is based on and claims the benefit of U.S. provisional patent application Ser. No. 62/156,096 filed May 1, 2015, and U.S. provisional patent application Ser. No. 62/156,082 filed May 1, 2015, and U.S. provisional patent application Ser. No. 62/156,089 filed May 1, 2015, the content of which is hereby incorporated by reference in its entirety.

BACKGROUND

Database systems are currently in wide use. In general, a database system includes a server that interacts with a data storage component to store data (and provide access to it) in a controlled and ordered way.

In one example, a database system includes a plurality of data centers, each having one or more servers. The data centers can be multi-tenant data centers that host data or services or both for a plurality of different tenants. Each tenant can correspond to, for example, a different organization.

The data centers can be disparately located from one another, for instance in different geographic regions. In some scenarios, it may be that data from a first data center is migrated to a second data center.

The discussion above is merely provided for general background information and is not intended to be used as an aid in determining the scope of the disclosed subject matter.

SUMMARY

Data is moved from a source system to a target system for a set of tenants. As soon as movement of key datasets is detected, the set of tenants are re-directed to the target system. Requests for non-key datasets are directed back to the source system until all datasets are moved.

This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used as an aid in determining the scope of the claimed subject matter. The claimed subject matter is not limited to implementations that solve any or all disadvantages noted in the background.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of one example of a data storage architecture.

FIG. 2 is a flow diagram of one example of the operation of architecture 100 in isolating data and moving data from a source data center to a target data center.

FIGS. 2A and 2B show a flow diagram illustrating one example of the operation of the architecture shown in FIG. 1 in moving data from a source container to a target container, in more detail.

FIG. 3 is a flow diagram showing one example of the operation of the architecture shown in FIG. 1 redirecting users (or tenants) from the source data center to the target data center.

FIG. 4 shows an example of a cloud computing architecture.

2

FIGS. 5-7 show examples of mobile devices that can be used in any of the architectures shown in previous figures.

FIG. 8 is a block diagram of a computing environment that can be used in any of the architectures shown in the previous figures.

DETAILED DESCRIPTION

FIG. 1 is a block diagram of various components of one illustrative data storage architecture 100. Data storage architecture 100 illustratively includes a source computing system 102 that interacts with user computing systems 103 and 105 to generate user interface displays 104-106 with user input mechanisms for interaction by users 108-110 (who may be tenants in a multi-tenant system, for instance). Users 108-110 interact with the user input mechanisms to control and manipulate system 102.

Architecture 100 can also include data isolation system 112, data move system 114 and target computing system 116, along with temporary secure storage system 118. Source computing system 102 illustratively includes application component 120, servers or processors 122, multi-tenant data in data store 124 and data to be moved in source container 126. It can include other items 131 as well. The data to be moved is illustratively data that is isolated from data in store 124 and can be broken into key datasets 128 and other (or non-essential) datasets 130. Data isolation system 112 illustratively includes tenant identification and tagging component 133, migration batching component 135, data isolation component 139 and processors or servers 141. Data move system 114 illustratively includes computing instance generator 132, key data notifier 134, user redirection system 136, target provisioning component 137, data destruction system 138, servers or processors 151, difference volume identifier 140, compute instances (such as instances 153, 155 and 157) that are generated by computing instance generator 132, and it can include other items 142 as well. Target computing system 116 illustratively includes application component 144, servers or processors 146, target container 148 and it can include other items 147 as well.

The computing instances (or compute instances) are illustratively different virtual machines (or different sets of virtual machines) that are configured to perform the functions indicated. They can have different sizes based on the physical resources reserved to them (such as the quantity of memory, disk space, processors or cores, etc.).

By way of overview, application components 120 and 144 illustratively run applications or services on systems 102 and 116, respectively. When tenant data (or any portion of data) is to be transferred from source system 102 to target system 116, data isolation system 112 isolates that data into source container 126. It can also identify key datasets 128 based on their metadata, their frequency of use, or based on a wide variety of other criteria. Data move system 114 provisions the tenants to be moved to target system 116 and begins to move the data from source container 126 to target container 148. When the key datasets 128 have been successfully moved, key data notifier 134 notifies redirection component 136, which redirects users 108-110 the users (or tenants) of the data being moved) to be serviced by target computing system 116 and target container 148. Any user requests for other datasets 130 are illustratively redirected back to source computing system 102, until those datasets are moved as well, at which point data destruction component destroys the data in source container 126. These operations will now be described in more detail.

FIG. 2 is a flow diagram illustrating one example of the operation of the architecture shown in FIG. 1 in isolating data to be moved from source computing system 102 to target computing system 116. In one example, data move system 114 first receives an input indicating that a tenant move is to be performed. This is indicated by block 40 in FIG. 2.

There may be a wide variety of different reasons that a given user or tenant, or a given set of users or tenants, are to have their data (which can include data, metadata, content, applications, services, etc.) moved from one data center to another. For instance, when a new data center is added, it may be that, for load rebalancing purposes, a set of tenants are to be moved to the data center. In another example, where a data center is added in a new geographic location, it may be that a group of tenants will be moved to that data center because they are physically located closer to it. Tenant data may be migrated because of data sovereignty laws, or for a wide variety of other reasons as well. Adding a data center is indicated by block 42 in FIG. 2, and moving for other reasons is indicated by block 44.

Data isolation system 112, and in particular tenant identification and tagging component 133, then identifies tenants that are to be moved. This is indicated by block 46 in FIG. 2. For instance, component 133 can identify tenants based upon their physical location (such as a country code that identifies the country where they reside). This is indicated by block 48. Component 132 can identify tenants based upon their billing location, as indicated by block 50, or based on a wide variety of other tenant-level property or attribute values. This is indicated by block 52.

Once the tenants are identified based upon the given criteria, component 132 tags the identified tenants with a unique tag. This is indicated by block 54 in FIG. 2. The tag can identify the particular relocation path for the tenant (such as indicating that the tagged tenant is to move from source computing system 102 to target computing system 116). This is indicated by block 56. The tag can include a wide variety of other information as well, or different information, as indicated by block 58.

Migration batching component 135 then performs any tenant batching that may be used by architecture 100. This is indicated by block 60. For example, where a relatively large number of tenants are to be moved, the batches may be formed based upon desired batch size. This is indicated by block 62. The batch size may be set to enhance the experience of the tenants or other end users that are being moved. By way of example, if the batch is very large, this may result in a less favorable user experience. However, if the batch is relatively small, this may result in the user experience being uninterrupted (or nearly uninterrupted). The batches can be generated on a variety of other batching criteria as well, and this is indicated by block 64.

Once the batches are identified, migration batching component 135 assigns a batch identifier to each of the tenants in the multi-tenant service, that are to be migrated. This is indicated by block 66.

Data isolation component 139 then selects a batch ID and isolates tenant data for tenants with the selected batch ID, into a separate database (such as isolated source container 126). Selecting a batch ID is indicated by block 68 in FIG. 2, and isolating the tenant data based on the selected batch ID is indicated by block 70.

It should be noted that the tenant data can take a wide variety of different forms. For instance, it can be generally high level tenant metadata 72, such as metadata that identifies the tenant, characteristics of the tenant, etc. It can also

be user identification and authorization data (such as roles, permissions, or a wide variety of other information for the various users in a tenant). This is indicated by block 74. In some cases, the actual content (such as documents or other content) may be stored on a different data store. In that case, the tenant data can include pointers to the content in that other data store. This indicated by block 76. The data can also include the content itself, as indicated by block 78, and various applications 80, functionality 82, services 83, or other information, data, services, etc. 84 that are hosted for a given tenant.

It may be that the identified tenants use other services as well, in architecture 100. In that case, tenant identification and tagging component 133 can also notify any other services that the given tenants are to be isolated for a move. This is indicated by block 86.

Data move system 114 then begins to move the batch of tenants to the target data center. This is indicated by block 88. In one example, target provisioning component 137 provisions (or otherwise sets up) the tenant, and a target container for the tenant, on the target data center. This is indicated by block 90.

System 114 then moves the data, and any pointers to content, for the tenant. Moving the data is indicated by block 92 and moving the pointers to content is indicated by block 94. System 114 can move other data or move data in other ways as well, and this is indicated by block 96. Moving the data is described in greater detail below with respect to FIGS. 2A and 2B.

User redirection system 136 then changes the computing system configuration to re-direct tenants to the target data center, once their data has been moved. This is indicated by block 97 in FIG. 2 and is described in greater detail below with respect to FIG. 3.

Data destruction component 138 then cleans up the source data center (or source computing system) 102 by deleting the data, once it has been moved. This is indicated by block 98.

Data move system 114 then determines whether there are any more batches of tenants to be moved. If so, processing reverts to block 68. Determining whether any additional batches are to be moved is indicated by block 99 in FIG. 2.

FIGS. 2A and 2B show a flow diagram illustrating the operation of architecture 100 in moving data, and ensuring that all the data has been moved from the source container to the target container. Data move system 114 first identifies the source and target containers 126 and 148. This is indicated by block 150. It is assumed that data isolation system 112 has already identified the data to be moved and isolated it into its own data container (or its own set of data containers) 126, which has no other tenant data (or other data that is not to be moved to system 116) in it. This is indicated by block 152. This can be done in other ways as well, as indicated by block 154.

Computing instance generator 132 then launches a first computing system instance that only has enumeration rights to source container 126. The first instance then enumerates all data inside container 126 and generates an enumeration list 156. Launching the first computing instance and enumerating the contents of source container 126 is indicated by blocks 156 and 158, respectively.

The list is stored in temporary secure storage system 118. System 118 is illustratively in a physically separate location from source computing system 102, as indicated by block 160. The enumeration list 156 illustratively has no indication that it relates to the environment of source computing

system **102**. It can be made in other ways as well. This is indicated by blocks **162** and **164**.

Once the enumeration list is stored in storage system **118**, computing instance generator **132** launches a second computing system instance that has read access to source container **126** and write access to target container **148**. This is indicated by block **166**. It reads the secure enumeration list **156** and copies data from the source container **126** to the target container **148** based on the enumeration list. This is indicated by blocks **168** and **170**.

Computing instance generator **132** then generates a third computing instance that has enumeration access to both source container **126** and target container **148**. It performs a full enumeration of both containers and compares them to generate a difference list, which now becomes the new enumeration list of items to be moved. The difference list will illustratively represent changes made to the tenant data in data container **126**, since the secure enumeration list **156** was created. Launching the third computing instance, performing the full enumeration and storing the difference list in the secure store is indicated by blocks **172**, **174** and **176**, respectively.

Difference volume identifier **140** then determines whether the volume of the differences (e.g., the number or size of items in the difference enumeration list) meets a given threshold. This is indicated by block **178**. If not, processing reverts to block **166** where the migration continues without interrupting the operation of source container **126**, with respect to its users **108-110**.

The threshold is illustratively set low enough that the subsequent migration of the remaining data will last for a sufficiently short time that the source container **126** can be placed in read only mode, without a significant, negative impact on users **108-110** of the source container. If the volume of differences meets the threshold, then source container **126** is placed in read only mode so that no further changes can be made to it. Placing it in read only mode is indicated by block **180**.

A computing instance performs a final enumeration of the source and target containers **126** and **148** to identify a final enumeration list, and a final copy of data is performed from source container **126** to target container **148**, based on the final enumeration list. This is indicated by block **182**. The application is then configured to point the users **108-110** of the data that was moved to target container **148**, and subsequent user requests are serviced by target computing system **116** and target container **148**. This is indicated by block **184**.

FIG. **3** is a flow diagram illustrating another example of the operation of data move system **114** in moving data and redirecting users. It is first assumed in FIG. **3** that an asynchronous move is being performed as discussed above. This is indicated by block **186**. In one example, when data is being moved as discussed above with respect to FIGS. **2A** and **2B** the computing instances enumerate the key datasets **128** and identify them as such. Thus, during movement of the data, the key datasets can be preferentially moved before the other datasets **130**. As mentioned above, the key datasets may be the datasets that are accessed by users to perform core functions. The key datasets can be other datasets or identified in other ways as well. They can be datasets most frequently or most recently accessed by users. They can be various portions of the tenant metadata, applications, services, functionality, content, etc.

As soon as all the key datasets **128** are moved, key data notifier **134** notifies the user redirection system **136**, and user redirection system **136** redirects the users **108-110** of the

data in source container **126** to target computing system **116** and target container **148**. This is indicated by blocks **188** and **190** in FIG. **3**. Data move system **114** continues to move non-key datasets from source container **126** to target container **148**. This is indicated by block **191**.

Once the key datasets are moved (even while the non-key datasets are still being moved), target system **116** processes user requests from target container **148**. This is indicated by block **192**.

In one example, users **108-110** may request other datasets **130**, which have not yet been moved to target container **148**. In such cases, user redirection system **136** redirects those requests back to source computing system **102** and source container **126**. This is indicated by block **194**. The user requests can be processed in other ways as well. This is indicated by block **196**.

When all datasets (both key datasets and non-key datasets) are copied to target container **148**, data destruction component destroys the source datasets in source container **126**. This is indicated by blocks **198** and **200**.

After that point, all user requests are serviced from target computing system **116** and target container **148**. This is indicated by block **202**.

It can thus be seen that the tenant being moved has very little disruption. There is a relatively short time window when the tenants data will be read only. Also, the data is transferred in a highly secure manner. Separation of computing instances with limited access rights greatly enhances security. Also, by initially isolating data of tenants to be moved into its own containers, efficiencies achieved by moving an entire container can be achieved as well.

The present discussion mentions a variety of different components. It will be noted that the components can be consolidated so that more functionality is performed by each component, or they can be divided so that the functionality is further distributed.

It should also be noted that the above discussion has shown one or more data stores. Each data store can be any of a wide variety of different types of data stores. Further, the data in the data store can be stored in multiple additional data stores as well. Also, the data stores can be local to the environments, agents, modules, and/or components that access them, or they can be remote therefrom and accessible by those environments, agents, modules, and/or components. Similarly, some can be local while others are remote.

The present discussion has mentioned processors and servers. In one embodiment, the processors and servers include computer processors with associated memory and timing circuitry, not separately shown. They are functional parts of the systems or devices to which they belong and are activated by, and facilitate the functionality of the other components or items in those systems.

Also, user interface displays have been discussed. They can take a wide variety of different forms and can have a wide variety of different user actuatable input mechanisms disposed thereon. For instance, the user actuatable input mechanisms can be text boxes, check boxes, icons, links, drop-down menus, search boxes, etc. They can also be actuated in a wide variety of different ways. For instance, they can be actuated using a point and click device (such as a track ball or mouse). They can be actuated using hardware buttons, switches, a joystick or keyboard, thumb switches or thumb pads, etc. They can also be actuated using a virtual keyboard or other virtual actuators. In addition, where the screen on which they are displayed is a touch sensitive screen, they can be actuated using touch gestures. Also,

where the device that displays them has speech recognition components, they can be actuated using speech commands.

A number of data stores have also been discussed. It will be noted they can each be broken into multiple data stores. All can be local to the systems accessing them, all can be remote, or some can be local while others are remote. All of these configurations are contemplated herein.

Also, the figures show a number of blocks with functionality ascribed to each block. It will be noted that fewer blocks can be used so the functionality is performed by fewer components. Also, more blocks can be used with the functionality distributed among more components.

FIG. 4 is a block diagram of a cloud computing architecture 500. Cloud computing provides computation, software, data access, and storage services that do not require end-user knowledge of the physical location or configuration of the system that delivers the services. In various embodiments, cloud computing delivers the services over a wide area network, such as the internet, using appropriate protocols. For instance, cloud computing providers deliver applications over a wide area network and they can be accessed through a web browser or any other computing component. Software or components of computing architecture 100 as well as the corresponding data, can be stored on servers at a remote location. The computing resources in a cloud computing environment can be consolidated at a remote data center location or they can be dispersed. Cloud computing infrastructures can deliver services through shared data centers, even though they appear as a single point of access for the user. Thus, the components and functions described herein can be provided from a service provider at a remote location using a cloud computing architecture. Alternatively, they can be provided from a conventional server, or they can be installed on client devices directly, or in other ways.

The description is intended to include both public cloud computing and private cloud computing. Cloud computing (both public and private) provides substantially seamless pooling of resources, as well as a reduced need to manage and configure underlying hardware infrastructure.

A public cloud is managed by a vendor and typically supports multiple consumers using the same infrastructure. Also, a public cloud, as opposed to a private cloud, can free up the end users from managing the hardware. A private cloud may be managed by the organization itself and the infrastructure is typically not shared with other organizations. The organization still maintains the hardware to some extent, such as installations and repairs, etc.

In the example shown in FIG. 4, some items are similar to those shown in FIG. 1 and they are similarly numbered. FIG. 4 specifically shows that some or all components of architecture 100 can be located in cloud 502 (which can be public, private, or a combination where portions are public while others are private). Therefore, users 108 and/or 110 use user systems 103, 105 to access those components through cloud 502.

FIG. 4 also depicts another example of a cloud architecture. FIG. 4 shows that it is also contemplated that some elements of architecture 100 can be disposed in cloud 502 while others are not. For example, system 102 can be disposed outside of cloud 502, and accessed through cloud 502. In another example, system 116 can be disposed outside of cloud 502, and accessed through cloud 502. In another example, system 114 can be disposed outside of cloud 502, and accessed through cloud 502. In another example, system 118 can be disposed outside of cloud 502, and accessed through cloud 502. Regardless of where they are located, they can be accessed directly by systems 103-105, through

a network (either a wide area network or a local area network), they can be hosted at a remote site by a service, or they can be provided as a service through a cloud or accessed by a connection service that resides in the cloud. All of these architectures are contemplated herein.

It will also be noted that architecture 100, or portions of it, can be disposed on a wide variety of different devices. Some of those devices include servers, desktop computers, laptop computers, tablet computers, or other mobile devices, such as palm top computers, cell phones, smart phones, multimedia players, personal digital assistants, etc.

FIG. 5 is a simplified block diagram of one illustrative example of a handheld or mobile computing device that can be used as a user's or client's hand held device 16, in which the present system (or parts of it) can be deployed. FIGS. 6-7 are examples of handheld or mobile devices.

FIG. 5 provides a general block diagram of the components of a client device 16 that can run components of architecture 100 or that interacts with architecture 100, or both. In the device 16, a communications link 13 is provided that allows the handheld device to communicate with other computing devices and under some embodiments provides a channel for receiving information automatically, such as by scanning. Examples of communications link 13 include an infrared port, a serial/USB port, a cable network port such as an Ethernet port, and a wireless network port allowing communication through one or more communication protocols including General Packet Radio Service (GPRS), LTE, HSPA, HSPA+ and other 3G and 4G radio protocols, 1xrtt, and Short Message Service, which are wireless services used to provide cellular access to a network, as well as 802.11 and 802.11b (Wi-Fi) protocols, and Bluetooth protocol, which provide local wireless connections to networks.

Under other embodiments, applications or systems are received on a removable Secure Digital (SD) card that is connected to a SD card interface 15. SD card interface 15 and communication links 13 communicate with a processor 17 along a bus 19 that is also connected to memory 21 and input/output (I/O) components 23, as well as clock 25 and location system 27.

I/O components 23, in one embodiment, are provided to facilitate input and output operations. I/O components 23 for various embodiments of the device 16 can include input components such as buttons, touch sensors, multi-touch sensors, optical or video sensors, voice sensors, touch screens, proximity sensors, microphones, tilt sensors, and gravity switches and output components such as a display device, a speaker, and or a printer port. Other I/O components 23 can be used as well.

Clock 25 illustratively comprises a real time clock component that outputs a time and date. It can also, illustratively, provide timing functions for processor 17.

Location system 27 illustratively includes a component that outputs a current geographical location of device 16. This can include, for instance, a global positioning system (GPS) receiver, a LORAN system, a dead reckoning system, a cellular triangulation system, or other positioning system. It can also include, for example, mapping software or navigation software that generates desired maps, navigation routes and other geographic functions.

Memory 21 stores operating system 29, network settings 31, applications 33, application configuration settings 35, data store 37, communication drivers 39, and communication configuration settings 41. Memory 21 can include all types of tangible volatile and non-volatile computer-readable memory devices. It can also include computer storage media (described below). Memory 21 stores computer read-

able instructions that, when executed by processor 17, cause the processor to perform computer-implemented steps or functions according to the instructions. Similarly, device 16 can have a client system 24 which can run various client applications or client-side applications. Processor 17 can be activated by other components to facilitate their functionality as well.

Examples of the network settings 31 include things such as proxy information, Internet connection information, and mappings. Application configuration settings 35 include settings that tailor the application for a specific enterprise or user. Communication configuration settings 41 provide parameters for communicating with other computers and include items such as GPRS parameters, SMS parameters, connection user names and passwords.

Applications 33 can be applications that have previously been stored on the device 16 or applications that are installed during use, although these can be part of operating system 29, or hosted external to device 16, as well.

FIG. 6 shows one example in which device 16 is a tablet computer 600. In FIG. 6, computer 600 is shown with user interface display displayed on the display screen 602. Screen 602 can be a touch screen (so touch gestures from a user's finger 604 can be used to interact with the application) or a pen-enabled interface that receives inputs from a pen or stylus. It can also use an on-screen virtual keyboard. Of course, it might also be attached to a keyboard or other user input device through a suitable attachment mechanism, such as a wireless link or USB port, for instance. Computer 600 can also illustratively receive voice inputs as well.

Additional examples of devices 16 can be used, as well. Device 16 can be a feature phone, smart phone or mobile phone. The phone includes a set of keypads for dialing phone numbers, a display capable of displaying images including application images, icons, web pages, photographs, and video, and control buttons for selecting items shown on the display. The phone includes an antenna for receiving cellular phone signals such as General Packet Radio Service (GPRS) and 1xrtt, and Short Message Service (SMS) signals. In some embodiments, phone also includes a Secure Digital (SD) card slot that accepts a SD card.

The mobile device can be personal digital assistant (PDA) or a multimedia player or a tablet computing device, etc. (hereinafter referred to as a PDA). The PDA can include an inductive screen that senses the position of a stylus (or other pointers, such as a user's finger) when the stylus is positioned over the screen. This allows the user to select, highlight, and move items on the screen as well as draw and write. The PDA also includes a number of user input keys or buttons which allow the user to scroll through menu options or other display options which are displayed on the display, and allow the user to change applications or select user input functions, without contacting the display. Although not shown, The PDA can include an internal antenna and an infrared transmitter/receiver that allow for wireless communication with other computers as well as connection ports that allow for hardware connections to other computing devices. Such hardware connections are typically made through a cradle that connects to the other computer through a serial or USB port. As such, these connections are non-network connections. In one embodiment, mobile device also includes a SD card slot that accepts a SD card.

FIG. 7 shows that the phone can be a smart phone 71. Smart phone 71 has a touch sensitive display 73 that displays icons or tiles or other user input mechanisms 75. Mechanisms 75 can be used by a user to run applications, make calls, perform data transfer operations, etc. In general, smart

phone 71 is built on a mobile operating system and offers more advanced computing capability and connectivity than a feature phone.

Note that other forms of the devices 16 are possible.

FIG. 8 is one example of a computing environment in which architecture 100, or parts of it, (for example) can be deployed. With reference to FIG. 8, an example system for implementing some embodiments includes a general-purpose computing device in the form of a computer 810. Components of computer 810 may include, but are not limited to, a processing unit 820 (which can be any processors or servers in the previous Figures), a system memory 830, and a system bus 821 that couples various system components including the system memory to the processing unit 820. The system bus 821 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus. Memory and programs described with respect to FIG. 1 can be deployed in corresponding portions of FIG. 8.

Computer 810 typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by computer 810 and includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media is different from, and does not include, a modulated data signal or carrier wave. It includes hardware storage media including both volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computer 810. Communication media typically embodies computer readable instructions, data structures, program modules or other data in a transport mechanism and includes any information delivery media. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. Combinations of any of the above should also be included within the scope of computer readable media.

The system memory 830 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 831 and random access memory (RAM) 832. A basic input/output system 833 (BIOS), containing the basic routines that help to transfer information between elements within computer 810, such as during start-up, is typically stored in ROM 831. RAM 832 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 820. By way of example, and not limitation, FIG. 8

illustrates operating system **834**, application programs **835**, other program modules **836**, and program data **837**.

The computer **810** may also include other removable/non-removable volatile/nonvolatile computer storage media. By way of example only, FIG. **8** illustrates a hard disk drive **841** that reads from or writes to non-removable, nonvolatile magnetic media, and an optical disk drive **855** that reads from or writes to a removable, nonvolatile optical disk **856** such as a CD ROM or other optical media. Other removable/non-removable, volatile/nonvolatile computer storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory cards, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The hard disk drive **841** is typically connected to the system bus **821** through a non-removable memory interface such as interface **840**, and optical disk drive **855** are typically connected to the system bus **821** by a removable memory interface, such as interface **850**.

Alternatively, or in addition, the functionality described herein can be performed, at least in part, by one or more hardware logic components. For example, and without limitation, illustrative types of hardware logic components that can be used include Field-programmable Gate Arrays (FPGAs), Program-specific Integrated Circuits (ASICs), Program-specific Standard Products (ASSPs), System-on-a-chip systems (SOCs), Complex Programmable Logic Devices (CPLDs), etc.

The drives and their associated computer storage media discussed above and illustrated in FIG. **8**, provide storage of computer readable instructions, data structures, program modules and other data for the computer **810**. In FIG. **8**, for example, hard disk drive **841** is illustrated as storing operating system **844**, application programs **845**, other program modules **846**, and program data **847**. Note that these components can either be the same as or different from operating system **834**, application programs **835**, other program modules **836**, and program data **837**. Operating system **844**, application programs **845**, other program modules **846**, and program data **847** are given different numbers here to illustrate that, at a minimum, they are different copies.

A user may enter commands and information into the computer **810** through input devices such as a keyboard **862**, a microphone **863**, and a pointing device **861**, such as a mouse, trackball or touch pad. Other input devices (not shown) may include a joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit **820** through a user input interface **860** that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A visual display **891** or other type of display device is also connected to the system bus **821** via an interface, such as a video interface **890**. In addition to the monitor, computers may also include other peripheral output devices such as speakers **897** and printer **896**, which may be connected through an output peripheral interface **895**.

The computer **810** is operated in a networked environment using logical connections to one or more remote computers, such as a remote computer **880**. The remote computer **880** may be a personal computer, a hand-held device, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer **810**. The logical connections depicted in FIG. **8** include a local area network (LAN) **871** and a wide area network (WAN) **873**, but may also include other networks. Such

networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

When used in a LAN networking environment, the computer **810** is connected to the LAN **871** through a network interface or adapter **870**. When used in a WAN networking environment, the computer **810** typically includes a modem **872** or other means for establishing communications over the WAN **873**, such as the Internet. The modem **872**, which may be internal or external, may be connected to the system bus **821** via the user input interface **860**, or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer **810**, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, FIG. **8** illustrates remote application programs **885** as residing on remote computer **880**. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

It should also be noted that the different embodiments described herein can be combined in different ways. That is, parts of one or more embodiments can be combined with parts of one or more other embodiments. All of this is contemplated herein.

Example 1 is a computing system, comprising:

a data move system configured to copy datasets to be moved from a source container, on a multi-tenant source computing system that runs an application, to a target container on a target computing system, the datasets to be moved comprising datasets for a set of identified tenants;

a key data notifier configured to detect a key dataset indicator, indicating that key datasets, which are a predefined subset of the datasets for the set of identified tenants, have been copied from the source container to the target container by the data move system, and generate a key dataset notification; and

a user re-direction system configured to, in response to the key dataset notification, configure the application to re-direct the set of tenants to the target computing system.

Example 2 is the computing system of any or all previous examples wherein the data move system is configured to continue to copy non-key datasets for the set of tenants from the source container to the target container, after the user re-direction system configures the application to re-direct the set of tenants to the target computing system.

Example 3 is the computing system of any or all previous examples wherein the user re-direction system is configured to detect a data access request, from a given tenant in the set of tenants, for a non-key dataset, and direct the data access request for the non-key dataset, from the given tenant, back to the source container on the source computing system.

Example 4 is the computing system of any or all previous examples wherein the user re-direction system detects that all datasets for the set of tenants have been copied from the source container to the target container and configures the application to service all data access requests for the set of tenants from the target container at the target computing system.

Example 5 is the computing system of any or all previous examples and further comprising:

a data destruction component configured to destroy the datasets for the set of tenants at the source container in the source computing system after the user re-direction system configures the application to service all data access requests for the set of tenants from the target container at the target computing system.

Example 6 is the computing system of any or all previous examples wherein the source computing system comprises a

13

source multi-tenant data center and the target computing system comprises a target multi-tenant data center, and further comprising:

a data isolation system configured to detect an input indicative of a desired move of the set of tenants from the source multi-tenant data center to the target multi-tenant data center, and isolate the datasets for the set of tenants, from datasets for other tenants, into the source container.

Example 7 is the computing system of any or all previous examples wherein the data isolation system is configured to detecting the input indicative of a desired move by detecting that the target data center is being added in a different geographic region from the source data center, and identifying the set of tenants based on a geographic location associated with tenants in the set of tenants.

Example 8 is the computing system of any or all previous examples and further comprising:

a target provisioning component configured to provision the set of tenants, and the target container, at the target multi-tenant data center.

Example 9 is a computer implemented method, comprising:

copying datasets to be moved from a source container, on a multi-tenant source computing system that runs an application, to a target container on a target computing system, the datasets to be moved comprising datasets for a set of identified tenants;

detecting a key dataset indicator, indicating that key datasets, which are a predefined subset of the datasets for the set of identified tenants, have been copied from the source container to the target container; and

in response to detecting the key dataset indicator, configuring the application to re-direct the set of tenants to the target computing system.

Example 10 is the computer implemented method of any or all previous examples and further comprising:

after configuring the application to re-direct the set of tenants to the target computing system, continuing to copy non-key datasets for the set of tenants from the source container to the target container.

Example 11 is the computer implemented method of any or all previous examples and further comprising:

detecting a data access request, from a given tenant in the set of tenants, for a non-key dataset; and

directing the data access request for the non-key dataset, from the given tenant, to the source container on the source computing system.

Example 12 is the computer implemented method of any or all previous examples and further comprising:

detecting that all datasets for the set of tenants has been copied from the source container to the target container; and configuring the application to service all data access requests for the set of tenants from the target container at the target computing system.

Example 13 is the computer implemented method of any or all previous examples and further comprising:

after configuring the application to service all data access requests for the set of tenants from the target container at the target computing system, destroying the datasets for the set of tenants at the source container in the source computing system.

Example 14 is the computer implemented method of any or all previous examples wherein the source computing system comprises a source multi-tenant data center and the target computing system comprises a target multi-tenant data center, and wherein copying comprises:

14

detecting an input indicative of a desired move of the set of tenants from the source multi-tenant data center to the target multi-tenant data center; and

isolating the datasets for the set of tenants, from datasets for other tenants, into the source container.

Example 15 is the computer implemented method of any or all previous examples wherein detecting an input comprises:

detecting that the target data center is being added in a different geographic region from the source data center; and identifying the set of tenants based on a geographic location associated with tenants in the set of tenants.

Example 16 is the computer implemented method of any or all previous examples wherein copying comprises:

provisioning the set of tenants, and the target container, at the target multi-tenant data center.

Example 17 is the computer implemented method of any or all previous examples wherein the predefined subset of datasets that comprise the key datasets comprises datasets that are predefined based on a frequency with which they are accessed by the set of tenants.

Example 18 is the computer implemented method of any or all previous examples wherein the predefined subset of datasets that comprise the key datasets comprises datasets that are predefined based on a recency with which they have been accessed by the set of tenants.

Example 19 is a computing system, comprising:

a data move system configured to copy datasets to be moved from a source container, on a multi-tenant source computing system that runs an application, to a target container on a target computing system, the datasets to be moved comprising datasets for a set of identified tenants;

a key data notifier configured to detect a key dataset indicator, indicating that key datasets, which are a predefined subset of the datasets for the set of identified tenants, have been copied from the source container to the target container by the data move system, and generate a key dataset notification; and

a user re-direction system configured to, in response to the key dataset notification, configure the application to re-direct the set of tenants to the target computing system, wherein the data move system is configured to continue to copy non-key datasets for the set of tenants from the source container to the target container, after the user re-direction system configures the application to re-direct the set of tenants to the target computing system, and wherein the user re-direction system is configured to detect a data access request, from a given tenant in the set of tenants, for a non-key dataset, and direct the data access request for the non-key dataset, from the given tenant, back to the source container on the source computing system.

Example 20 is the computing system of any or all previous examples wherein the user re-direction system is configured to detect that all datasets for the set of tenants have been copied from the source container to the target container and configures the application to service all data access requests for the set of tenants from the target container at the target computing system, and further comprising:

a data destruction component configured to destroy the datasets for the set of tenants at the source container in the source computing system after the user re-direction system configures the application to service all data access requests for the set of tenants from the target container at the target computing system.

Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in

15

the appended claims is not necessarily limited to the specific features or acts described above. Rather, the specific features and acts described above are disclosed as example forms of implementing the claims and other equivalent features and acts are intended to be within the scope of the claims.

What is claimed is:

1. A computing system, comprising:
 - a data move system configured to copy datasets to be moved from a source container, on a multi-tenant source computing system that runs an application, to a target container on a target computing system, the datasets to be moved comprising datasets for a set of identified tenants;
 - a key data notifier configured to detect a key dataset indicator, indicating that key datasets, which are a predefined subset of the datasets for the set of identified tenants, have been copied from the source container to the target container by the data move system, and generate a key dataset notification; and
 - a user re-direction system configured to, in response to the key dataset notification, configure the application to re-direct the set of tenants to the target computing system.
2. The computing system of claim 1 wherein the data move system is configured to continue to copy non-key datasets for the set of tenants from the source container to the target container, after the user re-direction system configures the application to re-direct the set of tenants to the target computing system.
3. The computing system of claim 2 wherein the user re-direction system is configured to detect a data access request, from a given tenant in the set of tenants, for a non-key dataset, and direct the data access request for the non-key dataset, from the given tenant, back to the source container on the source computing system.
4. The computing system of claim 3 wherein the user re-direction system detects that all datasets for the set of tenants have been copied from the source container to the target container and configures the application to service all data access requests for the set of tenants from the target container at the target computing system.
5. The computing system of claim 4 and further comprising:
 - a data destruction component configured to destroy the datasets for the set of tenants at the source container in the source computing system after the user re-direction system configures the application to service all data access requests for the set of tenants from the target container at the target computing system.
6. The computing system of claim 1 wherein the source computing system comprises a source multi-tenant data center and the target computing system comprises a target multi-tenant data center, and further comprising:
 - a data isolation system configured to detect an input indicative of a desired move of the set of tenants from the source multi-tenant data center to the target multi-tenant data center, and isolate the datasets for the set of tenants, from datasets for other tenants, into the source container.
7. The computing system of claim 6 wherein the data isolation system is configured to detecting the input indicative of a desired move by detecting that the target data center is being added in a different geographic region from the source data center, and identifying the set of tenants based on a geographic location associated with tenants in the set of tenants.

16

8. The computing system of claim 6 and further comprising:
 - a target provisioning component configured to provision the set of tenants, and the target container, at the target multi-tenant data center.
9. A computer implemented method, comprising:
 - copying datasets to be moved from a source container, on a multi-tenant source computing system that runs an application, to a target container on a target computing system, the datasets to be moved comprising datasets for a set of identified tenants;
 - detecting a key dataset indicator, indicating that key datasets, which are a predefined subset of the datasets for the set of identified tenants, have been copied from the source container to the target container; and
 - in response to detecting the key dataset indicator, configuring the application to re-direct the set of tenants to the target computing system.
10. The computer implemented method of claim 9 and further comprising:
 - after configuring the application to re-direct the set of tenants to the target computing system, continuing to copy non-key datasets for the set of tenants from the source container to the target container.
11. The computer implemented method of claim 10 and further comprising:
 - detecting a data access request, from a given tenant in the set of tenants, for a non-key dataset; and
 - directing the data access request for the non-key dataset, from the given tenant, to the source container on the source computing system.
12. The computer implemented method of claim 11 and further comprising:
 - detecting that all datasets for the set of tenants has been copied from the source container to the target container; and
 - configuring the application to service all data access requests for the set of tenants from the target container at the target computing system.
13. The computer implemented method of claim 12 and further comprising:
 - after configuring the application to service all data access requests for the set of tenants from the target container at the target computing system, destroying the datasets for the set of tenants at the source container in the source computing system.
14. The computer implemented method of claim 9 wherein the source computing system comprises a source multi-tenant data center and the target computing system comprises a target multi-tenant data center, and wherein copying comprises:
 - detecting an input indicative of a desired move of the set of tenants from the source multi-tenant data center to the target multi-tenant data center; and
 - isolating the datasets for the set of tenants, from datasets for other tenants, into the source container.
15. The computer implemented method of claim 14 wherein detecting an input comprises:
 - detecting that the target data center is being added in a different geographic region from the source data center; and
 - identifying the set of tenants based on a geographic location associated with tenants in the set of tenants.
16. The computer implemented method of claim 14 wherein copying comprises:
 - provisioning the set of tenants, and the target container, at the target multi-tenant data center.

17

17. The computer implemented method of claim 9 wherein the predefined subset of datasets that comprise the key datasets comprises datasets that are predefined based on a frequency with which they are accessed by the set of tenants.

18. The computer implemented method of claim 9 wherein the predefined subset of datasets that comprise the key datasets comprises datasets that are predefined based on a recency with which they have been accessed by the set of tenants.

19. A computing system, comprising:

a data move system configured to copy datasets to be moved from a source container, on a multi-tenant source computing system that runs an application, to a target container on a target computing system, the datasets to be moved comprising datasets for a set of identified tenants;

a key data notifier configured to detect a key dataset indicator, indicating that key datasets, which are a predefined subset of the datasets for the set of identified tenants, have been copied from the source container to the target container by the data move system, and generate a key dataset notification; and

a user re-direction system configured to, in response to the key dataset notification, configure the application to re-direct the set of tenants to the target computing

18

system, wherein the data move system is configured to continue to copy non-key datasets for the set of tenants from the source container to the target container, after the user re-direction system configures the application to re-direct the set of tenants to the target computing system, and wherein the user re-direction system is configured to detect a data access request, from a given tenant in the set of tenants, for a non-key dataset, and direct the data access request for the non-key dataset, from the given tenant, back to the source container on the source computing system.

20. The computing system of claim 19 wherein the user re-direction system is configured to detect that all datasets for the set of tenants have been copied from the source container to the target container and configures the application to service all data access requests for the set of tenants from the target container at the target computing system, and further comprising:

a data destruction component configured to destroy the datasets for the set of tenants at the source container in the source computing system after the user re-direction system configures the application to service all data access requests for the set of tenants from the target container at the target computing system.

* * * * *